

6.1 Fragen zur Vorlesung (30P)

6.1.1 MPI-IO-Implementierung

1. Welche Zugriffsarten gibt es auf die Daten einer Datei vom Standpunkt der Optimierung des parallelen Zugriffs?
2. Welche dieser Zugriffsarten erlauben dem darunterliegenden parallelen Dateisystem die größten Optimierungen?
3. Wie funktioniert Data Sieving beim Lesen?
4. Wie funktioniert Data Sieving beim Schreiben?
5. Welche Probleme gibt es bei Data Sieving?
6. Wie funktioniert Two-Phase I/O?
7. Welche MPI-IO-Hints kontrollieren bei ROMIO Data Sieving?
8. Welche MPI-IO-Hints kontrollieren bei ROMIO Two-Phase I/O?
9. Welche MPI-IO-Hints kontrollieren PVFS?

6.2 MPI-IO (360 Punkte)

Bisher wurde das traditionelle I/O-Konzept verwendet, um das Checkpointing und die Visualisierung zu betreiben. Nun stellte sich heraus, dass dies schlecht skaliert. Für ein Cluster mit 1.000+ Knoten ist das Checkpointing und die Visualisierung sehr wichtig. Glücklicherweise haben Sie festgestellt, dass in MPI-2 eine API für I/O bereitgestellt wird (MPI-IO). Hierbei kann jeder Prozess I/O durchführen, so dass keine Daten mehr zu einem Master-Prozess transferiert werden müssen. Schreiben Sie Ihr bisheriges Programm so um, dass MPI-IO-Aufrufe genutzt werden; verwenden Sie **keine** POSIX-Funktionen mehr in ihrem Code. Wahlweise können Sie auch mit dem bereitgestellten Programm-Skelett (s. Blatt 5) die Implementation von Vorne beginnen.

Um später die Leistung der Zugriffsvarianten vergleichen zu können, implementieren sie dies einmal mit kollektiver I/O und einmal mit individueller I/O (wenn möglich im selben Code).

6.2.1 Hinweise

Testen Sie das Programm auf Korrektheit und verwenden Sie mindestens die verschiedenen Prozesszahlen 1, 2 und 5. Testen Sie auch mindestens 2 verschiedene Größen der Matrix. Protokollieren Sie die dabei gemessenen Laufzeiten – entsprechen diese Ihren Erwartungen?

6.2.2 Abgabe

Die Abgabe sollte wie folgt strukturiert sein:

- Keine Binär- oder Objekt-Dateien mit abgeben!
- Ein Makefile um das Program direkt mit `make` übersetzen zu können.

- Der Quelltext soll in einem .tar.gz-Archiv (<nachname_1><nachname_2>...<nachname_k>.tar.gz) gepackt werden.
- Das Archiv soll nur einen Ordner mit dem Quellcode enthalten, der ebenfalls so benannt ist: <nachname_1><nachname_2>...<nachname_k>. Für die Gruppe Hans Mustermann und Klara Musterfrau würde der Ordner also MustermannMusterfrau heißen.
- Hinweis: Das Archiv kann mittels `tar -czf <archiv> <ordner>` erstellt werden.
- Alle Dateien und Ordner sollen keine Sonderzeichen (Leerzeichen, Umlaute etc.) enthalten.
- P.S.: eine umbenannte .zip-Datei ist KEINE .tar.gz-Datei.

Bei Nichteinhaltung der Abgaberichtlinien wird Ihre Abgabe von Dev Null korrigiert. :-)

Bearbeitungszeit			
Schwierigkeit	<input type="checkbox"/> zu leicht	<input type="checkbox"/> genau richtig	<input type="checkbox"/> zu schwer
Lehrreich	<input type="checkbox"/> wenig	<input type="checkbox"/> etwas	<input type="checkbox"/> sehr
Verständlichkeit	<input type="checkbox"/> großteils unklar	<input type="checkbox"/> teilweise unklar	<input type="checkbox"/> verständlich
Kommentar:			